A Scalable, Full-Custom Memory Design in CMOS 0.18 µm

Karl Leboeuf

Supervisor: Dr. Roberto Muscedere

Research Centre for Integrated Microsystems
University of Windsor

Outline

- Introduction
- Problem Outline
- Full-Custom RALUT Architecture
- Full-Custom Architecture Implementation
- Simulation Results
- HDL and Standard Cells vs. a Full-Custom Design
- Chip Design Goals and Methodology
- Additional Improvements
- Conclusion

Introduction to Range Addressable Look Up Tables

- Look up tables
  - A memory architecture where each storage element has a unique address
- Range Addressable Look Up Tables (RALUTs)
  - A non-linear memory storage element
  - Stored values are matched to a range of addresses
  - Each address row performs the following comparison:
    - Address (N) ≤ A ≤ Address (N+1)
  - Only one comparison will match, enabling the correct output

Introduction to Range Addressable Look Up Tables

Standard LUT

Address	Data
0000	0
0001	0
0010	0
0011	4
0100	4
0101	4
0110	4
0111	4
1000	4
1001	7
1010	7
1011	7
1100	7
1101	7
1110	8
1111	8

RALUT

Address	Data
0000	0
	0
	0
0011	4
	4
	4
	4
	4
	4
1001	7
	7
	7
	7
	7
1110	8
	8

Problem Outline

- The primary motivation for developing RALUTs was to improve number conversion, addition, and subtraction in Multi-Dimensional Logarithmic Number System (MDLNS) and Double Base Number System (DBNS) processors
- RALUTs are used instead of standard look-up tables, as significant area savings can be achieved
- The problem:
  - The existing RALUT design was implemented in CMOS 0.35 µm, an older technology
  - The design was not fabricated; real-world performance data is unavailable
  - Can a full custom RALUT design beat a commercial logic synthesizer in terms of area, speed, power, or all three?

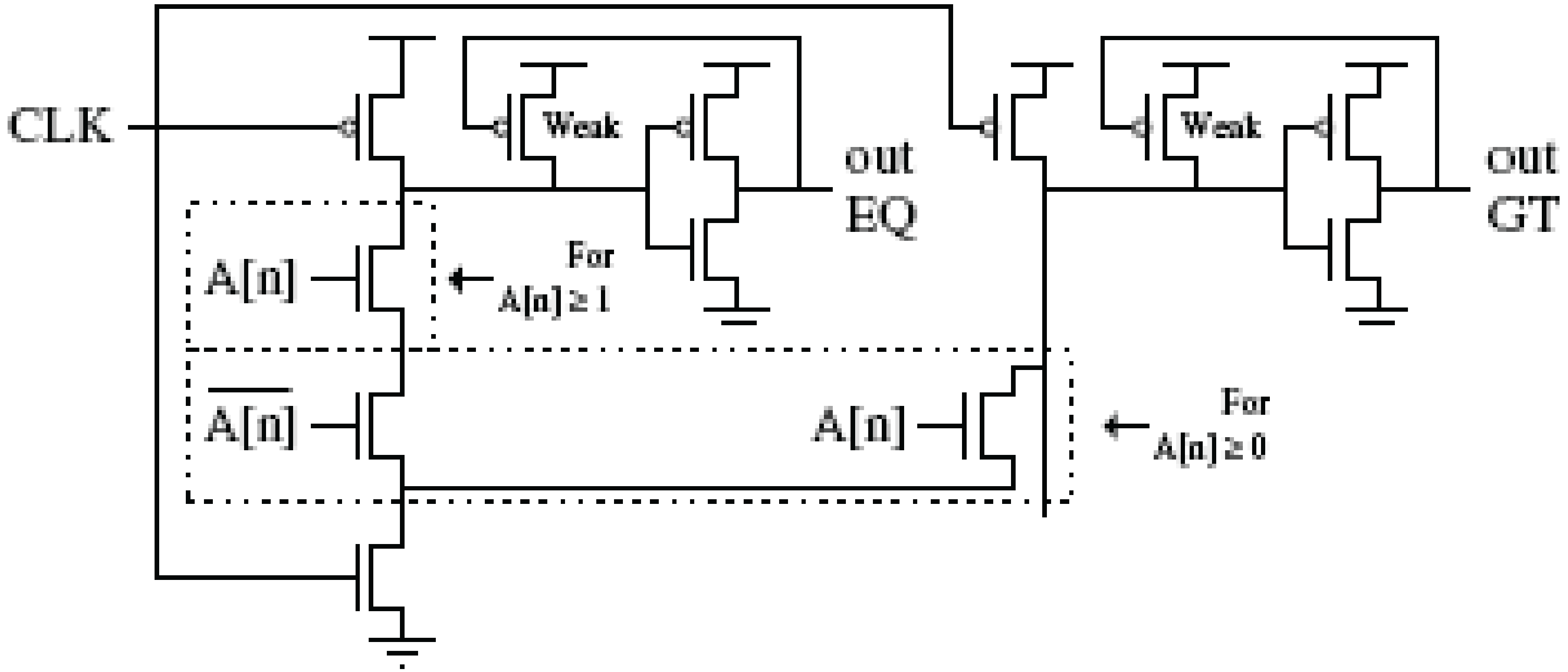
Full Custom RALUT Architecture: Overview

- RALUT designed using domino logic
- Domino logic
  - Significantly faster than static CMOS (~+70%)
  - Reduced area (~ -20%)
  - Sensitive to loading
  - Cells must be custom designed
- Design consists of the following elements:
  - Address decode circuit
  - Clock and input buffer rows
  - Output lines

Full Custom RALUT Architecture: Address Decode Circuit

- The address decode must compare the input address and determine if it is greater than or equal to each row address
- To eliminate problems arising from charge sharing, charge leakage, and long pre-charge time, small chains of NMOS transistors are used
- This breaks up the address decode into stages
- On the following slide is a schematic of the first decode stage

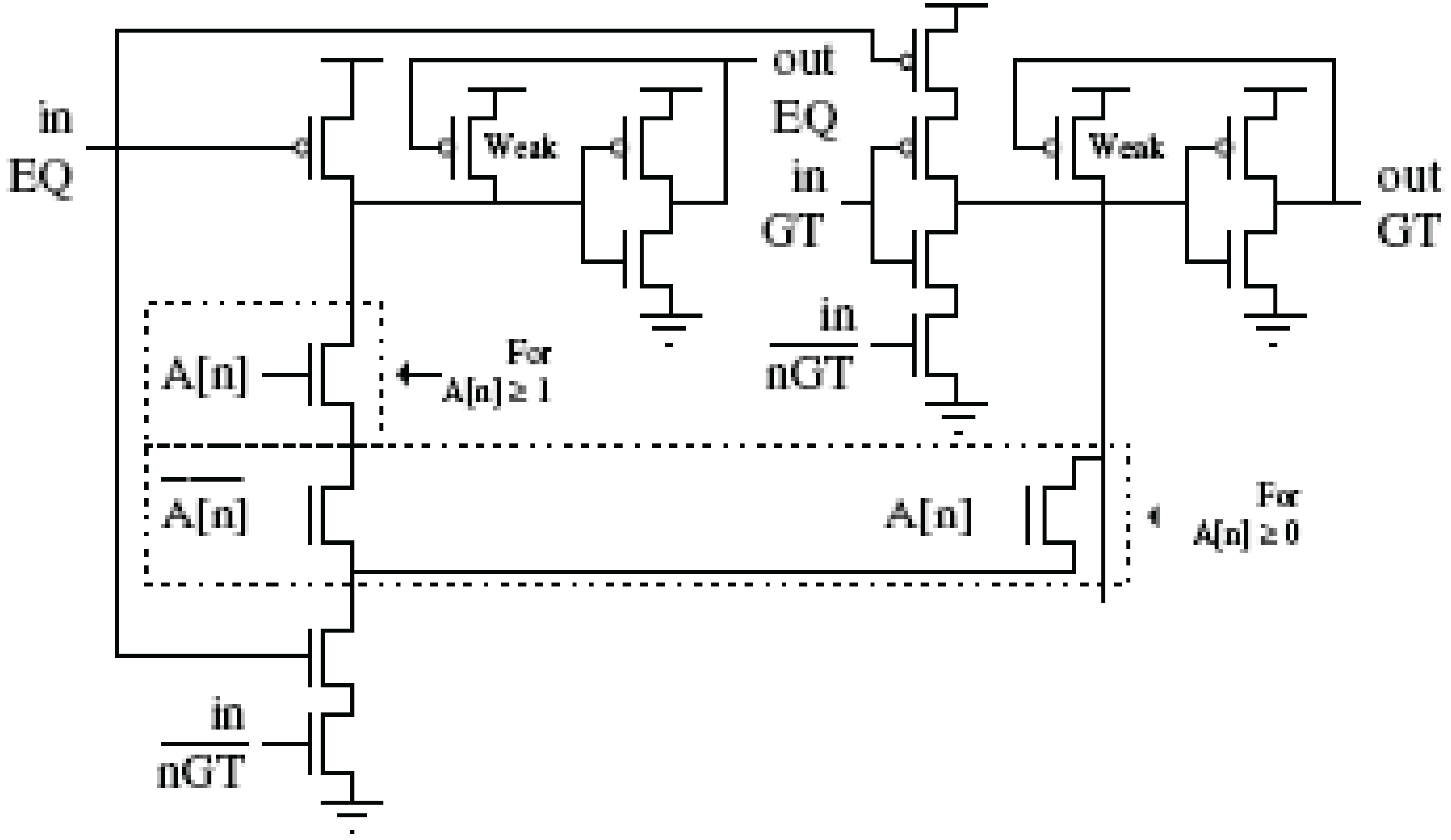
Full Custom RALUT Architecture: Address Decode Circuit



Full Custom RALUT Architecture: Address Decode Circuit

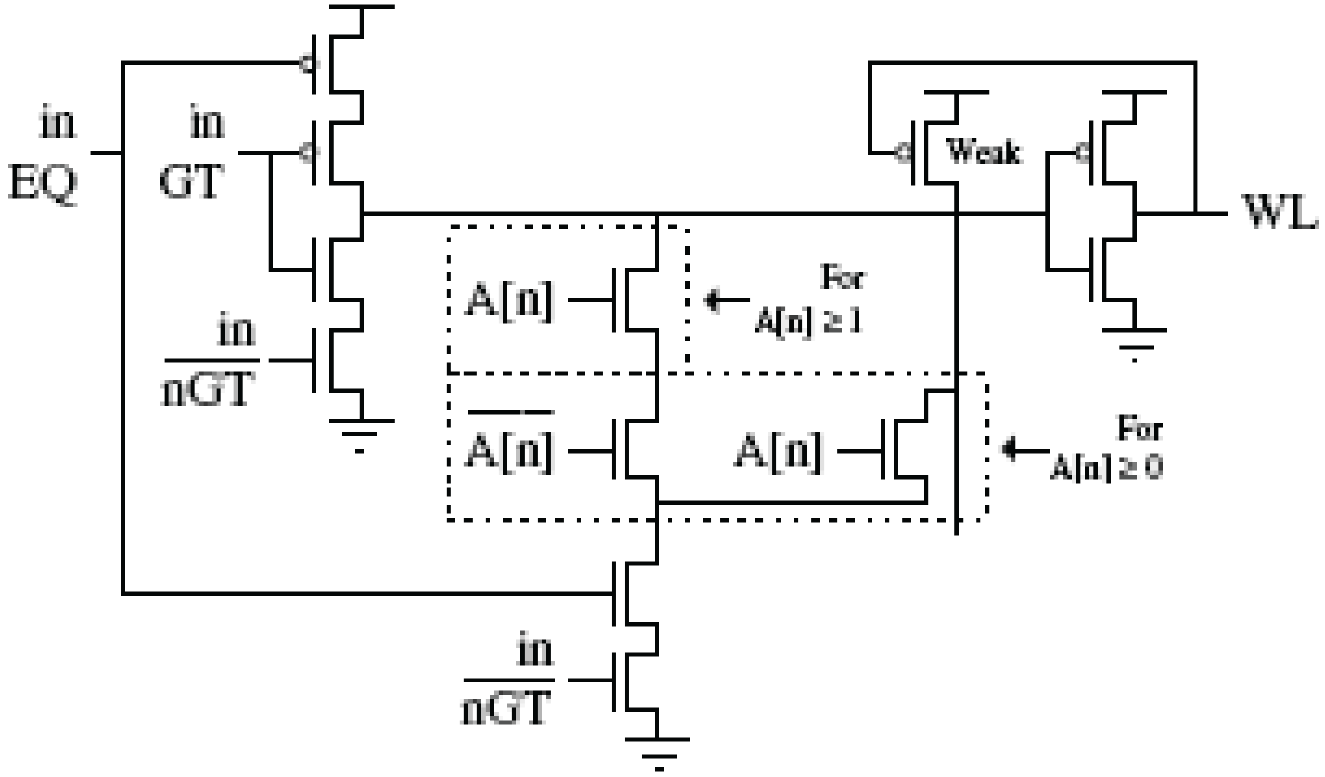
- As many “middle” stages as necessary are used to evaluate all but the first and last groups of bits
  - The clock signal only goes to the first stage to work the pre-charge and evaluate transistors
  - Subsequent pre-charge and evaluate transistors are controlled by the previous stage’s GT and EQ lines
  - This greatly reduces transistor switching, and by extension, power consumption
- It is pictured in the following slide

Full Custom RALUT Architecture: Address Decode Circuit



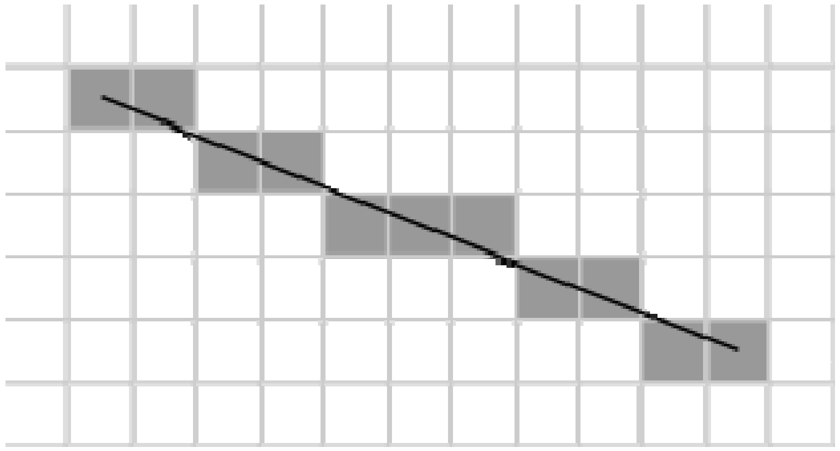
Full Custom RALUT Architecture: Address Decode Circuit

- The final address decode stage is shown below



Full Custom RALUT Architecture: Clock and Input Buffers

- Clock and input buffers occupy entire rows of the design
- The amount of clock and input buffering is easily scalable by changing a design parameter
- Placement of buffer rows is determined by the Bresenham line algorithm
  - Raster algorithm used to draw lines in MSPaint and many other applications
  - Allows buffer lines to be relatively evenly placed in non-power-of-two configurations



Full Custom RALUT Architecture: Output Lines

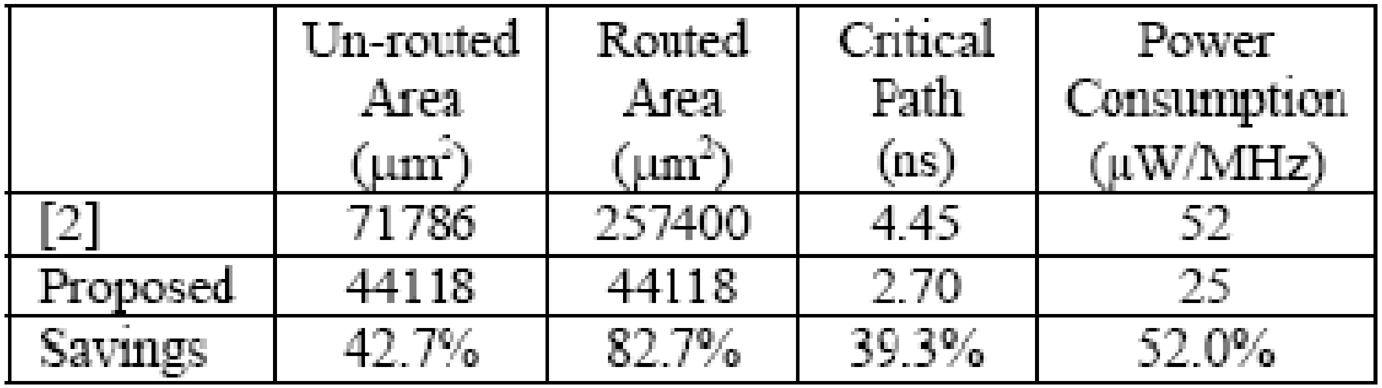
- The address decode circuit will enable a single output line
- Scalable design allows for any number of output bits

Full-Custom RALUT Design Implementation: Full-Custom P-Cells and SKILL Code

- All of the aforementioned circuit elements were designed in CMOS 0.35 µm, making heavy use of full-custom p-cells
- A p-cell is a parameterized design cell
  - Example of parameters include NMOS / PMOS transistor sizes, special even/odd row and first/last row settings
- SKILL code is the programming language used by the Cadence Design Framework II environment
  - SKILL code was developed to automatically parameterize and place all of the cells and IO pins

Advancing the State of the Art: Migration to CMOS 0.18 µm

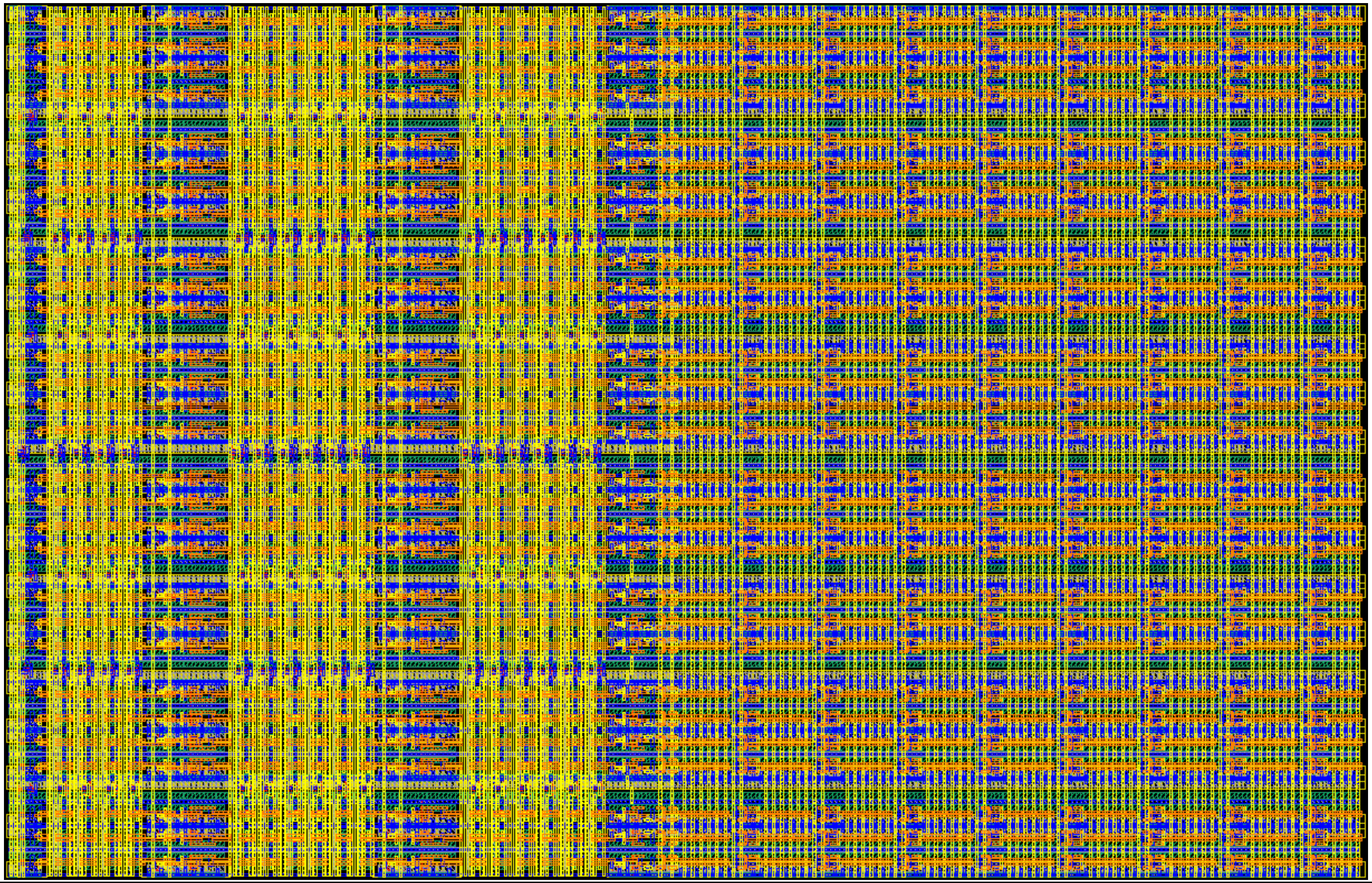
- First, the CMOS 0.35 µm design was ported to a more recent node; CMOS 0.18 µm
- All custom cells were rescaled by hand
  - Simple not an option, as many design rules changed between nodes
- Immediate area, speed, and power improvements
- **The following results are based on post place-and-route simulations comparing the 0.35 µm design to the new 0.18 µm design**



	Un-routed Area (μm^2)	Routed Area (μm^2)	Critical Path (ns)	Power Consumption (μW/MHz)
[2]	71786	257400	4.45	52
Proposed	44118	44118	2.70	25
Savings	42.7%	82.7%	39.3%	52.0%

0.18 μm RALUT Design

- 128 rows, 16 address bits, 64 output bits



Design Alternatives: HDL and Logic Synthesis

- As opposed to using a full-custom design, standard cell libraries may be used to implement RALUTs
- Standard cells employ static CMOS logic; they are slower and larger than domino, but far less sensitive to loading
- This reduces the complexity in performing logic synthesis and placement-and-routing

Design Alternatives: HDL and Logic Synthesis

- verilog code for the RALUT was written, synthesized, and tested
- Post-place-and-route results were compared to the custom cell RALUT
- Results highly dependant on the design size (number of rows, address bits, output bits)
- Larger designs can take from days to weeks to synthesize, requiring very large amounts of RAM
- Full-custom vs. verilog / standard-cell comparison:
  – Standard cell approach requires much less area for small designs than the RALUT
  – Standard cell design critical path delay and area varies wildly depending on the exact bit patterns used for addressing and outputs
  – Larger designs yield comparable area requirements
  – Simulations showed that the custom design was faster for medium/large designs

Real World Performance: Chip Design and Fabrication

- No “real data” on performance is available for the full-custom or standard cell designs
- Must fabricate a chip comparing designs

Chip Design Goals

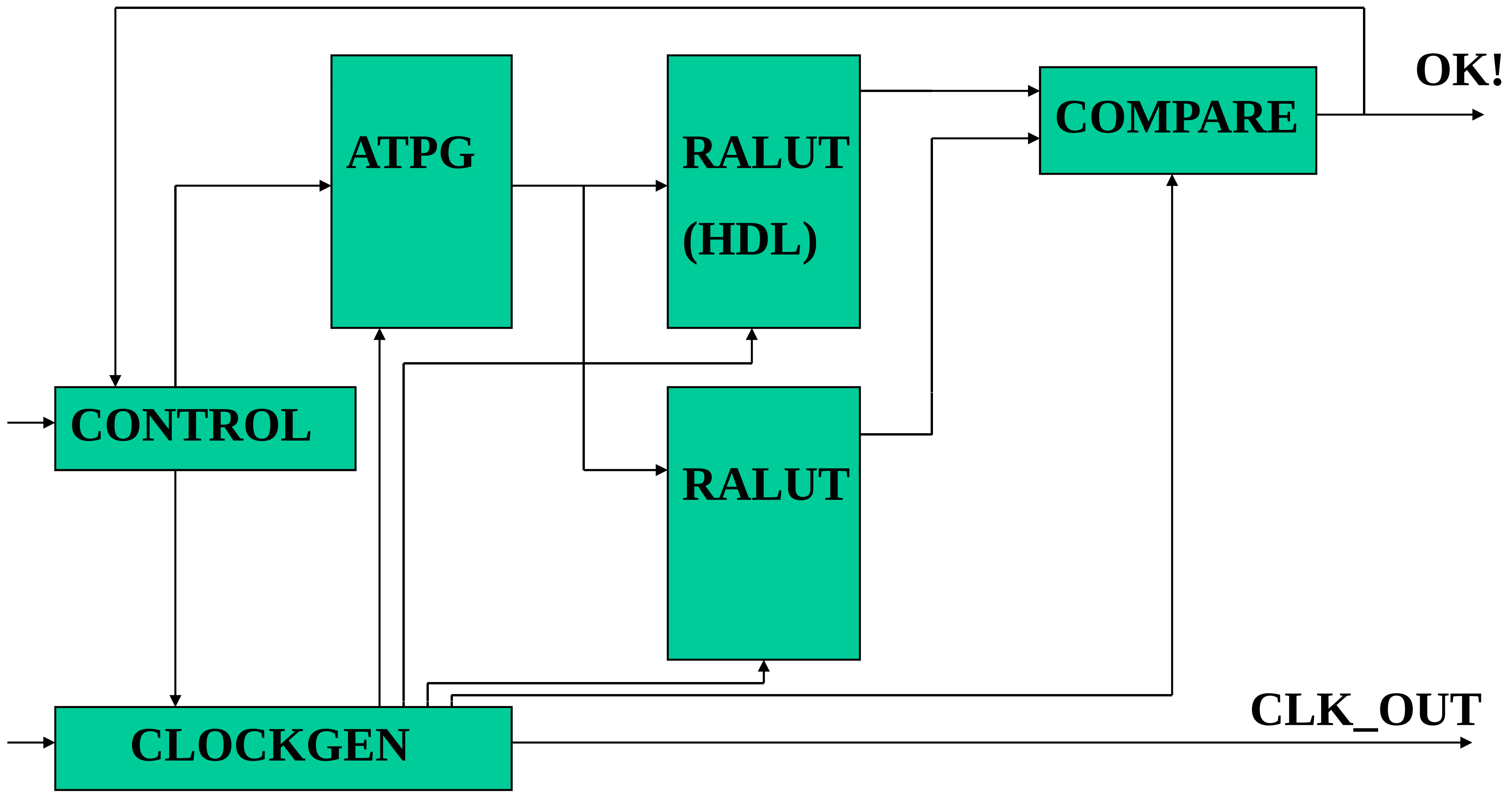
- Determine maximum (correct) operating frequency of both designs
- Determine average power consumption
- Perform a fair comparison on the same chip, where both designs are subject to the same process variation effects and temperature

Chip Design Considerations and Parameters

- 128 rows were used, this was the largest number of rows that could be accommodated by the available design space
- 16 address bits and 64 output bits were used
- Random bit patterns were used for the address and output bit patterns

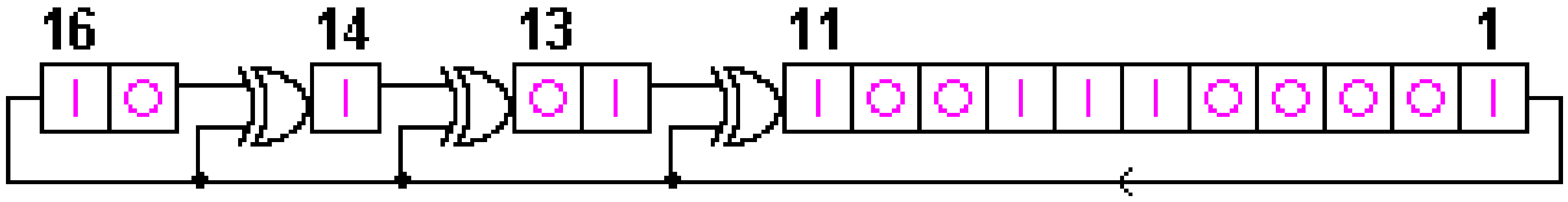
- As few I/O pins as possible are desired; this design will be highly I/O bound
- Simple interfacing with a generic microcontroller or FPGA will be used to configure the chip, and read the output
  - The design will use 8 input pins, 8 output pins, as well as some additional pins for status signals and external clocks

Chip Design: Overview



Chip Design: Automatic Test Pattern Generator (ATPG) Block

- The automatic test pattern generator (ATPG) block generates the address bits for both RALUT designs
- It is desirable that all the input bits switch randomly to approximate typical use
- A counter cannot be used because higher order bits do not switch as often as low order bits
- The ATPG employs a linear feedback shift register (LFSR) to generate “better”, pseudo-random bit patterns
- The LFSR will cycle through all 65535 possible input patterns



Chip Design: Clock Block

- The “Clock” block in the chip overview performs several functions
  - Selects between an external clock, and an internal ring-oscillator clock
  - Selects which clock signal is sent to the clock output pin
  - Selects the number of inverters to use in the ring-oscillator
  - If desired, scales the clock signal by 2, 4, 8, 16, or 32 by using a counter
  - Disables the clock to various parts of the chip to assist in estimating power consumption

Chip Design: Control Block

- The controller is a finite state machine which uses an external, low-speed clock
- The chip's 8-bit input bus goes directly to the controller
  - Control words and instructions are loaded using this bus

- It communicates with, and configures, the ATPG, CLOCK, and places data on the output bus
- If the compare fails between the custom and standard cell designs, the controller immediately halts the clock so that the failing system can be identified

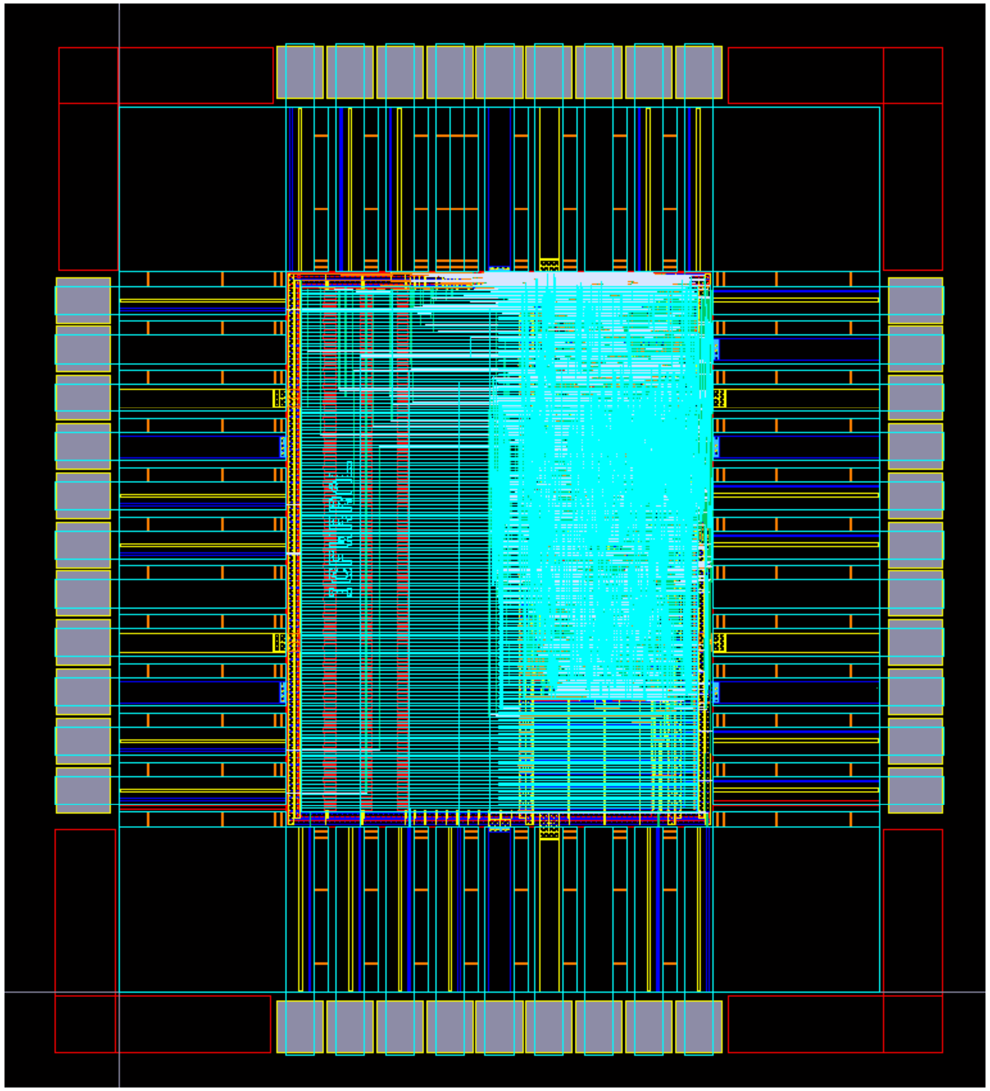
Chip Design Methodology

- The following steps were carried out in creating the chip
  - verilog code was written for the ATPG, controller, clock, comparator, the HDL version of the RALUT, and other components not shown, such as registers and glue logic
  - The verilog code was synthesized using Synopsys, synthesis parameters were chosen to yield the fastest design possible
  - A variable-frequency ring oscillator was designed
    - Works by selecting the number of inverters to include in the ring
    - Should provide a clock frequency between approximately 750MHz and 50 MHz (exact values depend significantly on PVT)

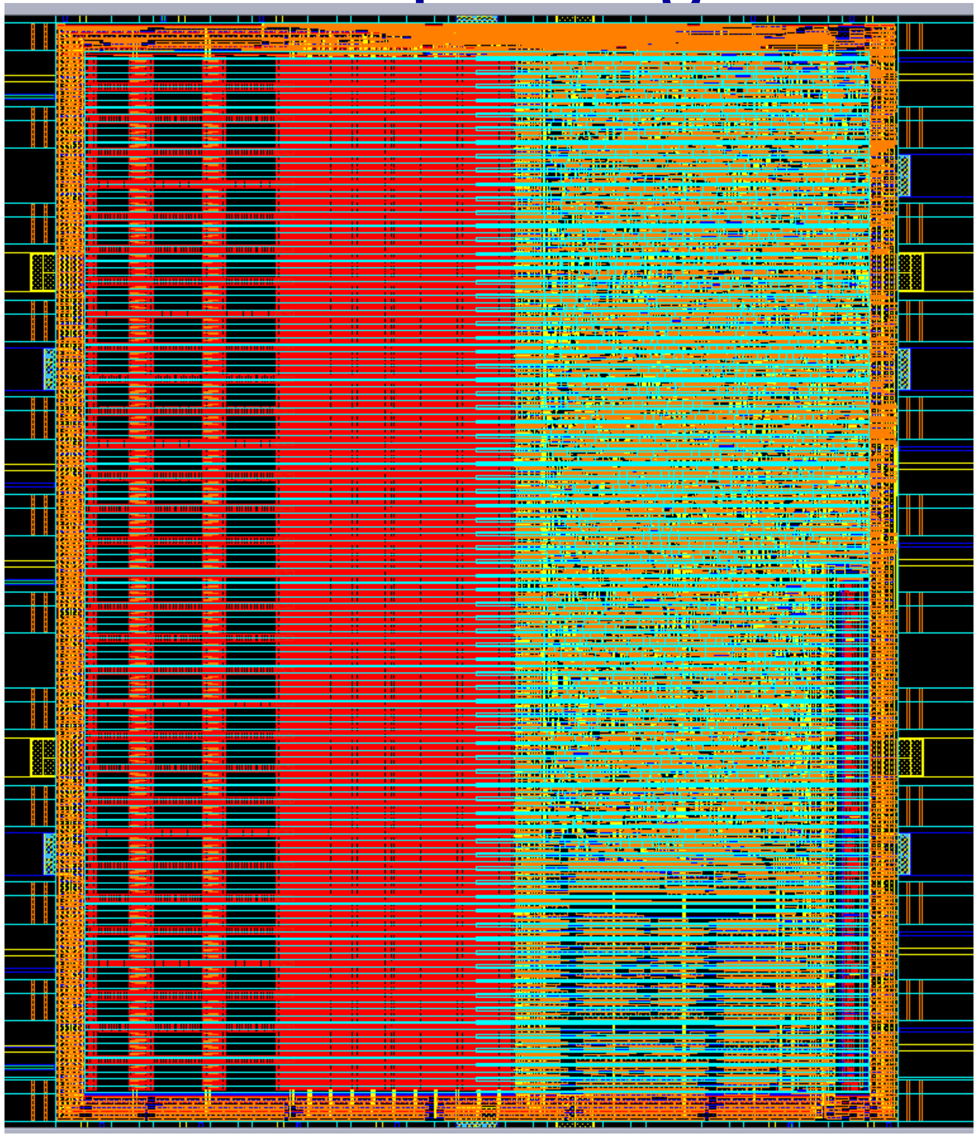
Chip Design Methodology

- The ring oscillator was extracted with parasitic capacitances, and simulated using Spectre
- An abstract view of the oscillator was created for use in Encounter, the place and route tool
- Extraction, simulation, and abstract generation was also performed on the full-custom RALUT
- Encounter was used to perform power ring and stripe placement, cell and macro placement and routing, clock-tree generation for the standard cell logic, and metal filling
- Final design was exported from Encounter, and DEFIN was used to import back to Virtuoso; final DRC checks were performed using Calibre
- Final design submitted to CMC for fabrication

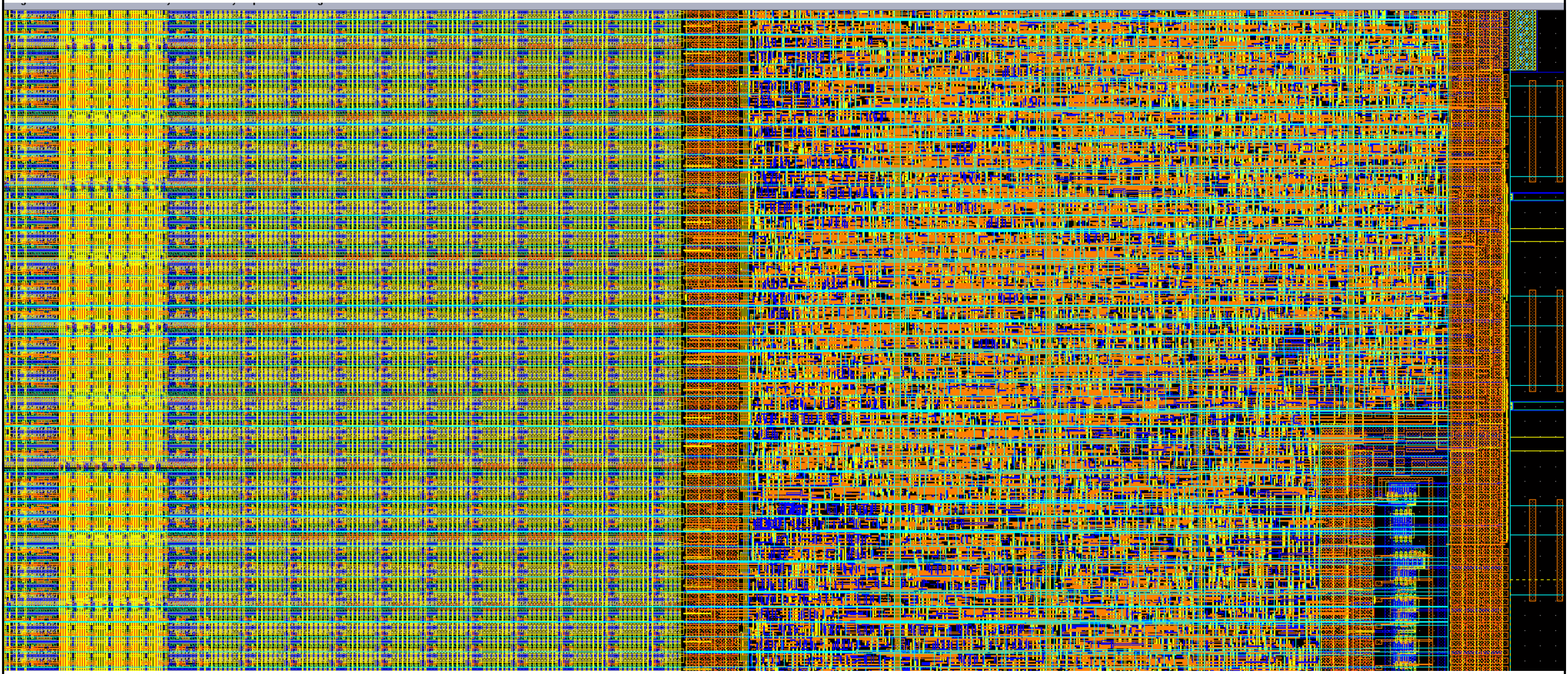
Final Chip Design



Final Chip Design



Final Chip Design



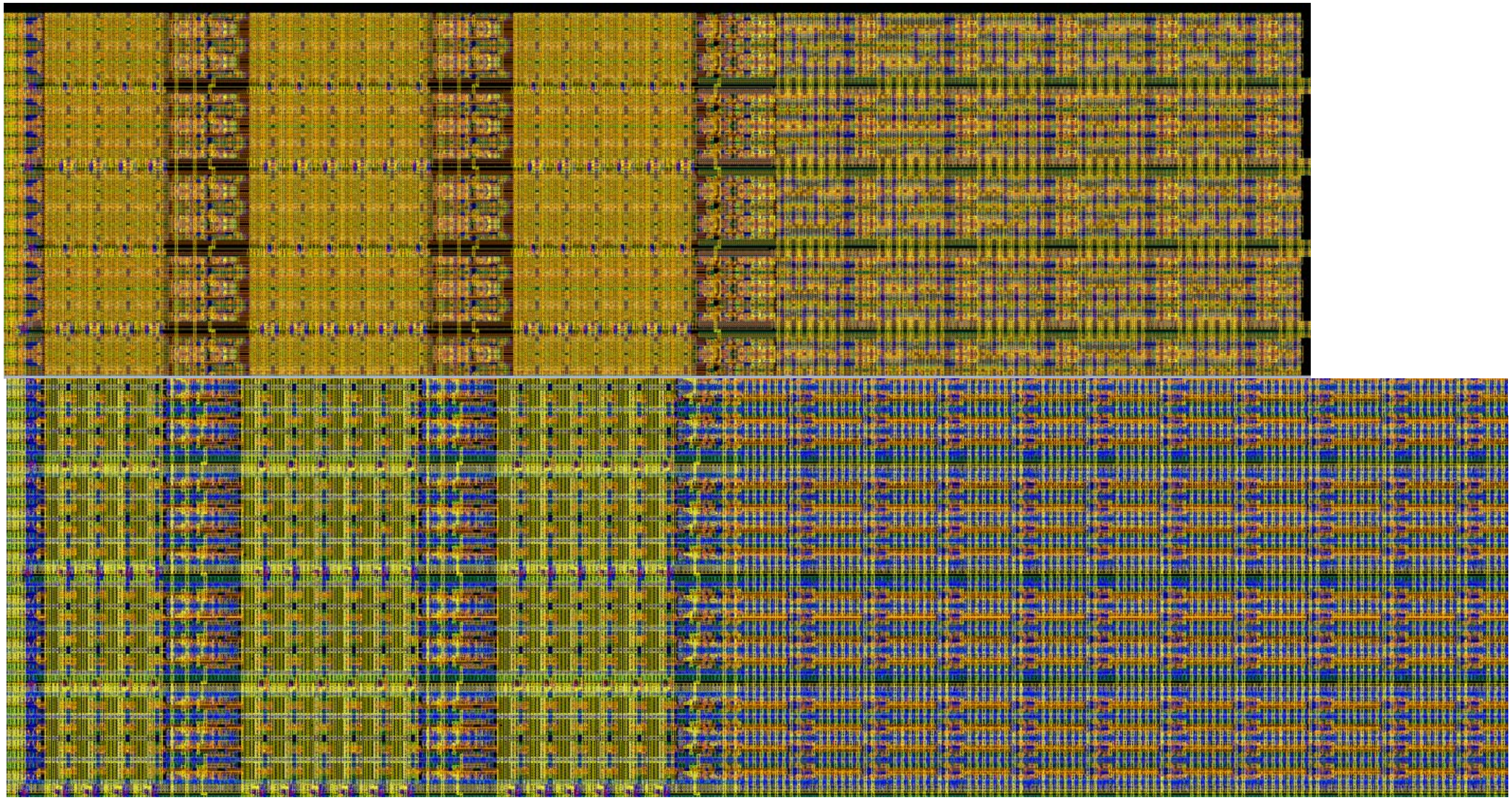
Chip Results

- Fabricated chip received at the end of February
- Testing scheduled for April

Additional Improvements: Area Reduction

- Several more improvements to reduce area are nearly complete
  - Power and ground rails are now superimposed over the design, create a significant height reduction
  - Output bits are now pushed closer together, reducing overall design width
  - Typical design area is reduced by approximately 30%
  - Operating frequency may be reduced due to increased parasitic capacitances

Additional Improvements: Area Reduction



Additional Improvements: Power Consumption

- Currently working on a pre-decode system
- Divide RALUT into a series of 2^N sub-RALUTs
- Would tentatively function by taking the N most significant address bits, and use them in a one-hot decoder scheme, enabling the clock for only one of the sub-tables
- This will greatly reduce power consumption, as well as aid in “squaring up” the design; currently most practical design parameters will implement a very tall and skinny table

Conclusions

- Once the chip is tested, real-world performance data will be available for this type of design
- Additional improvements to reduce area and power consumption are underway
- These advances will greatly enhance any system which can make use of RALUTs, such as MDLNS processors and non-linear function generation circuits
- The state of the art of RALUT design has been advanced significantly

References

1. Muscedere, R.; Leboeuf K., "A Dynamic Address Decode Circuit for Implementing Range Addressable Look-Up Tables "*IEEE International Symposium on Circuits and Systems (ISCAS)* 2008, Accepted February 2008.
2. R. Muscedere, "Difficult Operations in the Multidimensional Logarithmic Number System," PhD thesis, Univ. of Windsor, 2003.
3. R. Muscedere, V. Dimitrov, G. A. Jullien, and W. C. Miller, "Efficient techniques for binary-to-multidigit multidimensional logarithmic number system conversion using range-addressable look-up tables,"*Computers, IEEE Transactions on,* vol. 54, pp. 257-271, 2005.
4. P. Srivastava, A. Pua, and L. Welch, "Issues in the design of domino logic circuits," in *VLSI, 1998. Proceedings of the 8th Great Lakes Symposium on*, 1998, pp. 108-112.